I Erläuterungen

Voraussetzungen gemäß KCBG und Abiturerlassen BG jeweils in der für den Abiturjahrgang geltenden Fassung

Standardbezug
Die nachfolgend ausgewiesenen Kompetenzbereiche sind für die Bearbeitung der jeweiligen Aufgabe besonders bedeutsam. Darüber hinaus können weitere, hier nicht ausgewiesene Kompetenzbereiche für die Bearbeitung der Aufgabe nachrangig bedeutsam sein, zumal die Kompetenzbereiche in engem Bezug zueinanderstehen. Die Operationalisierung des Bezugs zu den Kompetenzbereichen des Standardbezugs erfolgt in Abschnitt II.

Aufgabe	Kompetenzbereiche				
	K1	K2	K3	K4	K5
1.1	X	X			
1.2	X	X			
1.3.1				X	
1.3.2				X	
1.3.3				X	
1.3.4				X	
1.4	X		X		
2.1	X			X	
2.2		X		X	
2.3.1	X				
2.3.2		X		X	
2.4.1				X	
2.4.2			X		
2.5.1	X		X		
2.5.2			X		

Inhaltlicher Bezug
Die nachfolgend ausgewiesenen Themenfelder sind die wesentliche inhaltliche Grundlage für die vorliegenden Aufgaben. Darüber hinaus können weitere, hier nicht explizit ausgewiesene Themenfelder für die Bearbeitung nachrangig bedeutsam sein.
Q1: Objektorientierte Softwareentwicklung
Q2: Datenbanksysteme
Q3: Datenkommunikation
verbindliche Themenfelder: Objektorientierte Modellierung (Q1.1), Implementierung von Klassen und Assoziationen (Q1.2), Konzeptionelle und logische Modellierung einer Datenbank (Q2.1), Datenabfrage und Datenmanipulation mit SQL (Q2.2), Serielle Kommunikation (Q3.1)

II Lösungshinweise

In den nachfolgenden Lösungshinweisen sind alle wesentlichen Gesichtspunkte, die bei der Bearbeitung der einzelnen Aufgaben zu berücksichtigen sind, konkret genannt und diejenigen Lösungswege aufgezeigt, welche die Prüflinge erfahrungsgemäß einschlagen werden. Selbstverständlich sind jedoch Lösungswege, die von den vorgegebenen abweichen, aber als gleichwertig betrachtet werden können, ebenso zu akzeptieren.

Aufg.	erwartete Leistungen	BE I	BE II	BE III
1.1	beschreiben, erklären Entitätstypen (grafische Darstellung als Rechteck) kategorisieren gleichartige Entitäten, die sich durch die gleichen Eigenschaften (Attribute) beschreiben lassen. Im vorliegenden Modell ist beispielsweise `Mitarbeiter` ein Entitätstyp. Attribute charakterisieren einen Entitätstyp bzw. einen Beziehungstyp. Die Attribute des Entitätstyps `Mitarbeiter` sind `mID`, `name`, `pw` und `beruf`. Attribute werden im ERM durch Ellipsen dargestellt. Identifizierende Attribute, so genannte Schlüsselattribute, dienen der eindeutigen Identifizierung von Entitäten eines Entitätstyps. Ein Primärschlüssel kann aus mehreren identifizierenden Attributen zusammengesetzt sein und wird durch Unterstreichung gekennzeichnet. Der Primärschlüssel von Mitarbeiter ist `mID`. Die Rauten stehen für Beziehungstypen. Durch Beziehungen (Relationships) werden die Abhängigkeiten zwischen Entitäten ausgedrückt. Ein Beziehungstyp ist die Abstraktion gleichartiger Beziehungen und kann ebenfalls Eigenschaften besitzen. Der Beziehungstyp `prüft` besitzt das Attribut `status`. Über die Kardinalität wird festgelegt, wie viele Entitäten einer Entitätsmenge mit Entitäten einer anderen Entitätsmenge in Beziehung stehen können oder müssen. Die Kardinalitäten im vorliegenden ERM sind in [min,max]-Notation dargestellt und bedeuten am Beispiel des Beziehungstyps `teil von`, dass jede Messkontrolle genau zu einem Jahresbericht gehört. Ein Jahresbericht besteht aus vier Messkontrollen. Die Mehrfachbeziehung `prüft` ist eine ternäre Beziehung. Jedes Messinstrument wird von einem bis beliebig vielen Mitarbeitern geprüft (muss-Beziehung). Ein Messinstrument gehört zu keiner bis beliebig vielen Messkontrollen. Jeder Mitarbeiter kann zu beliebig vielen Messkontrollen und -instrumenten gehören.			
	beschreiben	2		
	erklären		2	2

1.2	überführen
	`Jahresbericht(jID, jahr)`
	`Messkontrolle(kID, beginn, ende, jID#)`
	`Mitarbeiter(mID, vorname, name, beruf, pw)`
	`Prüfung(mID#, iID#, kID#, status)`
	`Messinstrument(iID, hersteller, geräteArt, garMessanzahl, aktiv, einheit)`
	`Messwert(wID, wert, timestamp, iID#)`
		6		

	begründen Alle Entitätstypen werden mit ihren Attributen in Relationen überführt. Der Name des Mitarbeiters muss zur Einhaltung der ersten Normalform aufgespalten werden. Die Mehrfachbeziehung `prüft` führt zu einer Beziehungsmengenrelation `Prüfung`, in der die PK der beteiligten Relationen zum FK werden. Eine 1:n-Beziehungen wird ohne eigene Relation realisiert, hier am Beispiel der Beziehung `ist Teil`. Der PK der 1-Relation wird FK in der n-Relation. Alle Nicht-Schlüsselattribute der Relationen sind vollständig vom PK abhängig und es existieren keine transitiven Abhängigkeiten, so dass die 3. Normalform eingehalten wird.		3	

Nicht für den Prüfling bestimmt

Aufg.	erwartete Leistungen	BE I	BE II	BE III

1.3.1	formulieren
	`UPDATE Messinstrument SET aktiv = false`
	`WHERE hersteller = 'Ondokei'`
	`AND geräteArt = 'temperatur';`
	
	`INSERT INTO Messinstrument(hersteller, geräteArt,`
	`            garMessanzahl, aktiv, einheit)`
	`VALUES('B+B', 'temperatur', 2000000, true, '° Celsius');`
	
	`INSERT INTO Messwerte(wert, timestamp, iID#)`
	`VALUES (10, NOW(), 352);`
			5	

1.3.2	entwickeln
	`SELECT iID, geräteArt`
	`FROM   Messinstrument i`
	`WHERE  aktiv = true AND garMessanzahl <=`
	`       (SELECT(COUNT(*)+ 60 * 24 * 7 * 6)`
	`       FROM Messwerte w WHERE i.iID = w.iID);`
			2	1

1.3.3	entwickeln
	`SELECT name, vorname, iID, geräteArt`
	`FROM Messkontrolle k`
	`      JOIN Prüfung p ON (k.kID = p.kID)`
	`      JOIN Mitarbeiter m ON (m.mID = p.mID)`
	`      JOIN Messinstrument i ON (i.iID = p.iID)`
	`WHERE p.status = 'laufend'`
	`AND (k.beginn BETWEEN '2023-01-01' AND '2023-03-31' OR`
	`     k.ende   BETWEEN '2023-01-01' AND '2023-03-31')`
	`ORDER BY m.name, m.vorname;`
			2	2

1.3.4	entwickeln
	`SELECT COUNT(DISTINCT DAY (timestamp))`
	`      AS 'Taganzahl für optimale Nutzung'`
	`FROM messwert m1`
	`WHERE YEAR(timestamp) = 2023`
	`AND MONTH(timestamp) = 02`
	`AND iID = 150`
	`AND(648000) <`
	`     (SELECT MAX(wert)`
	`      FROM messwert m2`
	`      WHERE DAY(m1.timestamp) = DAY(m2.timestamp))`
	`      AND m2.iID = 150;`
			1	3

Aufg.	erwartete Leistungen	BE I	BE II	BE III
1.4	entwickeln, zeichnen Team [1,n] gehört zu [1,m] Mitarbeiter Team [0,n] ist Teamleiter [1,1] Mitarbeiter Mitarbeiter [1,1] qualifiziert [0,n] visuelle Kontrol... Turbine (tID, anzStartvorgänge, status, wirkungsgrad) [5,5] prüft [1,n] visuelle Kontrol... Schleuse (sID, isGeschlossen) [3,3] prüft [1,n] visuelle Kontrol... visuelle Kontrol... (vID, beginn, ende) [2,4] teil von [1,1] Jahresbericht			
	entwickeln		3	3
	zeichnen	3		
	Summe 40	**11**	**18**	**11**

Aufg.	erwartete Leistungen	BE I	BE II	BE III
2.1	überführen, implementieren			

```java
public class Turbine {
  private int turbinenID;
  private int drehzahlOptimal;
  private int drehzahlAktuell;
  private double leitschaufelWinkel;
  private DateTime letzterStart;
  private long laufzeit;
  private boolean gesperrt;

  public Turbine(int turbinenID) {
    this.turbinenID = turbinenID;
    laufzeit = 0;
  }

  public void einschalten() {
    if(drehzahlAktuell == 0&& !gesperrt) {
      drehzahlAktuell = drehzahlOptimal;
      letzterStart = new DateTime();
    }
  }

  public void ausschalten() {
    if(drehzahlAktuell != 0) {
      drehzahlAktuell = 0;
      DateTime ende = new DateTime();
      laufzeit += ende.until(letzterStart);
    }
  }

  public void sperren() {
    gesperrt = true;
  }

  public void entsperren() {
    gesperrt = false;
  }

  public boolean isGesperrt() {
    return gesperrt;
  }
}
```

	überführen	4		
	implementieren		2	

Aufg.	erwartete Leistungen	BE		
		I	II	III
2.2	implementieren		4	4

```
public class Controller {
  private Serial serial;

  public Controller(String serialPort) {
    serial = new Serial(serialPort, 19200, 8, 2, 0);
  }

  public boolean schalte(Turbine turbine) {
    boolean ok = false;
    serial.write(CMD_CHANGEMODE_0 + "\n");
    if(serial.readLine().equals("ACK")) {
      String werte = " " + turbine.getTurbinenID() + ";" +
      turbine.getDrehzahlAktuell() + ";" +
      turbine.getLeitschaufelWinkel();
      serial.write(CMD_SET_STATUS + werte + "\n");
      if(serial.readLine().equals("ACK")) {
        ok = true;
      }
    }
    return ok;
  }

  public int lesePegelstand(char channel){
    byte wert = 0;
    if(channel == 'A') {
      serial.write(CMD_GET_CHANNEL_A + "\n");
      wert = serial.read();
    }
    if(channel == 'B') {
      serial.write(CMD_GET_CHANNEL_B + "\n");
      wert = serial.read();
    }
    return wert;
  }
}
```

Aufg.	erwartete Leistungen	BE		
		I	II	III
2.3.1	erläutern Sequenzdiagramme beschreiben die Kommunikation zwischen Objekten in einem bestimmten Szenarium. Es wird beschrieben, welche Objekte beteiligt sind, welche Informationen (Nachrichten) sie austauschen und in welcher zeitlichen Reihenfolge der Informationsaustausch stattfindet. Sequenzdiagramme enthalten eine implizite Zeitachse, die von oben nach unten verläuft. Die Reihenfolge der Pfeile in einem Sequenzdiagramm gibt die zeitliche Reihenfolge der Nachrichten an.		2	
	beschreiben Ein Objekt der Klasse `Steuerung` existiert bereits und wird im Diagramm als Rechteck dargestellt, an das sich als gestrichelte Linie die Lebenslinie anschließt. Der schmale Balken auf der Lebenslinie des Steuerung-Objektes, auch Aktivitätsbalken genannt, zeigt an, dass eine Methode des Objekts aktiv ist. Während dieser ersten Aktivierung werden drei Erzeugungsnachrichten (`<<create>>`) für die Objekte der Klassen `Controller`, `Rechenanlage` und `Stoerungsmeldung` ausgeführt und die `run`-Methoden der Threads jeweils durch die asynchrone Nachricht `start()` aufgerufen. Asynchrone Nachrichten haben eine offene Pfeilspitze. Asynchron bedeutet, dass der Aufrufer nicht wartet, sondern unmittelbar nach dem Aufruf fortfährt. In der `run()`-Methode des Rechenanlage-Objekts ist bis zur Beendigung der Wiederholung (`loop`) folgender Ablauf dargestellt: Die zwei Pegelstände werden mit dem Aufruf der synchronen Nachrichten (gefüllte Pfeilspitzen) des `Controller`-Objekts `lesePegelstand('A')` und `lesePegelstand('B')` abgerufen. Synchron bedeutet, dass der Aufrufer wartet, bis das aufgerufene Verhalten beendet wurde. Die Antwort auf einen synchronen Aufruf wird mit einer gestrichelten Linie und offener Pfeilspitze dargestellt, also hier die Rückgabe der ermittelten Messwerte. Optional (`opt`) wird daraufhin die Methode `schalten()` des Steuerungsobjekts aufgerufen, wenn sich die zuletzt ermittelte Wasserstandsstufe zur vorherigen geändert hat. Ist der Pegelstand vor dem Rechen mindestens 1 größer als hinter dem Rechen wird die Methode `starteReinigung()` aufgerufen. Hierbei sendet das Objekt eine Nachricht an sich selbst. Am Ende der Wiederholung wartet der Thread 60 Sekunden.	3	1	

Aufg.	erwartete Leistungen	BE I	BE II	BE III
2.3.2	implementieren (Code siehe unten)	2	3	3

```
public class Rechenanlage extends Thread {
  private int pegelstandVor;
  private int pegelstandNach;
  private int wasserstandStufe;
  public final static int NIEDRIGWASSER = 1;
  public final static int NORMAL = 2;
  public final static int HOCHWASSER = 3;
  private boolean running;
  private Steuerung steuerung;

  public Rechenanlage(Steuerung st) {
    steuerung = st;
    wasserstandStufe = 0;
    running = true;
  }

  public void run() {
    while(running) {
      pegelstandVor =
             steuerung.getController().lesePegelstand('A');
      pegelstandNach =
             steuerung.getController().lesePegelstand('B');
      int wasserstandAktuell = 0;
      if(pegelstandVor < 25) {
        wasserstandAktuell = NIEDRIGWASSER;
      } else if(pegelstandVor > 85) {
        wasserstandAktuell = HOCHWASSER;
      } else {
        wasserstandAktuell = NORMAL;
      }
      if(wasserstandAktuell != wasserstandStufe) {
        wasserstandStufe = wasserstandAktuell;
        steuerung.schalten(wasserstandStufe);
      }
      if(pegelstandVor - pegelstandNach >= 1 ) {
        starteReinigung();
      }
      Thread.sleep(60000);
    }
  }

  public void beenden() {
    running = false;
  }
}
```

Aufg.	erwartete Leistungen	BE I	BE II	BE III
2.4.1	implementieren	2	4	4

```java
public class Steuerung {
  private Turbine[] turbinen;
  private int schleusenStatus;
  private Stoerungsmeldung sm;
  private Rechenanlage ra;
  private Controller controller;

  public Steuerung(String serialPort, int drehzahlOptimal,
                   double leitschaufelWinkel) {
    turbinen = new Turbine[5];
    for(int i = 0; i < turbinen.length; i++) {
      turbinen[i] = new Turbine(i+1);
      turbinen[i].setDrehzahlOptimal(drehzahlOptimal);
      turbinen[i].setLeitschaufelWinkel(leitschaufelWinkel);
    }
    schleusenStatus = 0;
    controller  = new Controller(serialPort);
    if(controller.open()) {
      sm = new Stoerungsmeldung(this);
      sm.start();
      ra = new Rechenanlage(this);
      ra.start();
    }
  }

  public void  schalten(int wasserstandStufe) {
    if(wasserstandStufe == 1 || wasserstandStufe == 3) {
      for(int i = 0; i < turbinen.length; i++) {
        turbinen[i].ausschalten();
        controller.schalte(turbinen[i]);
      }

      if(wasserstandStufe == 3) {
        schleusenStatus = 1; //Schleusen geöffnet
        controller.schalteSchleusen(schleusenStatus);
      }
    } else  {
      if(schleusenStatus == 1) {
        schleusenStatus = 0; //Schleusen geschlossen
        controller.schalteSchleusen(schleusenStatus);
      }
      int lf = 0;
      for(int i = 0; i < turbinen.length && lf < 4; i++) {
        if(!turbinen[i].isGesperrt()) {
          turbinen[i].einschalten();
          controller.schalte(turbinen[i]);
          lf++;
        }
      }
    }
  }
}
```

Aufg.	erwartete Leistungen	BE		
		I	II	III
2.4.2	entwickeln, zeichnen **behandleStoerung(turbinenID: int)** index := turbinenID - 1 turbinen[index] sperren turbinen[index].drehzahlAktuell <> 0? (J / N) J: turbinen[index] ausschalten controller.schalte(turbinen[index]) turbinen[4] nicht gesperrt und turbinen[4] drehzahlAktuell = 0 ? (J / N) J: turbinen[4] einschalten controller.schalte(turbinen[index]) N: ∅ N: ∅			
	entwickeln		2	2
	zeichnen	2		

Aufg.	**erwartete Leistungen**	**BE**		
		I	**II**	**III**
2.5.1	entwickeln, zeichnen Anlagensteuerung anmelden abmelden Anlage hochfahren Betriebsart einstellen Durchflussrate ermitteln <<include>> Mitarbeiter Drehzahl und Winkel einstellen Messung erlauben Messwerte prüfen extension points Messwertabweichung <<extend>> Messerlaubnis entziehen Abweichung des Messwertes > 1% ? Anlage runterfahren			
	entwickeln zeichnen	 3	3	2

Aufg.	**erwartete Leistungen**	**BE**		
		I	**II**	**III**
2.5.2	modellieren, zeichnen **FernsteuerungsServer** – server : ServerSocket + FernsteuerungsServer(port : int, kwa : KraftwerksAnlage) + start() + abmelden() 1 – kraftwerksAnlage **KraftwerksAnlage** – betriebsArt : String + KraftwerksAnlage() + anmelden(name : String, passwort : String) : boolean + hochfahren() + runterfahren() + einstellenBetriebsart(art : String) + einstellenDrehzahl(drehzahl : int) + einstellenLeitschaufelWinkel(winkel : double) + ermittleMesswert(mi : Messinstrument) : int + pruefeMesswert(mi : Messinstrument) : boolean + erlaubeMessung(mi : Messinstrument) + entzieheMesserlaubnis(mi : Messinstrument) 1 – steuerung **Steuerung** * – messgeraete – mitarbeiter * 0..1 – angemeldet **Messinstrument** – iid : int – geraeteArt : String – hersteller : String – anzHerstellerGarantie : int – einheit : String – aktiv : boolean + Messinstrument(art : String, hersteller : String, einheit : String) **Mitarbeiter** – mid : int – name : String – beruf : String – passwort : String + Mitarbeiter(name : String) * – messwerte **Messwert** – datetime : DateTime – wert : double + Messwert(wert : double) : void			
	modellieren		3	3
	zeichnen	2		
	Summe 60	**18**	**24**	**18**

III Bewertung und Beurteilung

Die Bewertung und Beurteilung erfolgt unter Beachtung der nachfolgenden Vorgaben nach § 33 der Oberstufen- und Abiturverordnung (OAVO) in der jeweils geltenden Fassung. Bei der Bewertung und Beurteilung der sprachlichen Richtigkeit in der deutschen Sprache sind die Bestimmungen des § 9 Abs. 12 Satz 3 OAVO in Verbindung mit Anlage 9b anzuwenden.
Bei der Bewertung und Beurteilung der Übersetzungsleistung in den Fächern Latein und Altgriechisch sind die Bestimmungen des § 9 Abs. 14 OAVO in Verbindung mit Anlage 9c anzuwenden.

Der Fehlerindex ist nach Anlage 9b zu § 9 Abs. 12 OAVO zu berechnen. Für die Ermittlung der Punkte nach Anlage 9a zu § 9 Abs. 12 OAVO sowie Anlage 9c zu § 9 Abs. 14 OAVO wird jeweils der ganzzahlige nicht gerundete Prozentsatz bzw. Fehlerindex zugrunde gelegt.

Für die Bewertung in den modernen Fremdsprachen ist der „Erlass zur Bewertung und Beurteilung von schriftlichen Arbeiten in allen Grund- und Leistungskursen der neu beginnenden und fortgeführten modernen Fremdsprachen in der gymnasialen Oberstufe, dem beruflichen Gymnasium, dem Abendgymnasium und dem Hessenkolleg“ vom 7. August 2020 (ABl. S. 519) zugrunde zu legen. Demnach erfolgt die Bewertung und Beurteilung mit der Maßgabe, dass lediglich bei der Ermittlung des Prüfungsergebnisses (Note) aus Prüfungsteil 1 und 2 gerundet wird.

Darüber hinaus sind die Vorgaben der Erlasse „Hinweise zur Vorbereitung auf die schriftlichen Abiturprüfungen (Abiturerlass)“, „Hinweise zur Vorbereitung auf die schriftlichen Abiturprüfungen im beruflichen Gymnasium (fachrichtungs-/ schwerpunktbezogene Fächer) (Abiturerlass BG)“ und „Durchführungsbestimmungen zum Landesabitur“ in der für den Abiturjahrgang geltenden Fassung zu beachten.

Als Kriterien für die Bewertung und Beurteilung dienen unter Beachtung der Zielsetzung der gymnasialen Oberstufe nach § 1 Abs. 2 OAVO neben dem Inhaltlichen auch die in den Kerncurricula genannten überfachlichen Kompetenzen, insbesondere die Sprachkompetenz und Wissenschaftspropädeutik; dies zeigt sich u.a. in qualitativen Merkmalen wie Strukturierung, Differenziertheit, (fach-)sprachlicher Gestaltung und Schlüssigkeit der Argumentation.

Im Fach Praktische Informatik besteht die Prüfungsleistung aus der Bearbeitung eines Vorschlags, wofür insgesamt maximal 100 BE vergeben werden können. Ein Prüfungsergebnis von **5 Punkten (ausreichend)** setzt voraus, dass mindestens 45% der zu vergebenden BE erreicht werden. Ein Prüfungsergebnis von **11 Punkten (gut)** setzt voraus, dass mindestens 75% der zu vergebenden BE erreicht werden.

Gewichtung der Aufgaben und Zuordnung der Bewertungseinheiten zu den Anforderungsbereichen

Aufgabe	Bewertungseinheiten in den Anforderungsbereichen			Summe
	AFB I	AFB II	AFB III	
1	11	18	11	**40**
2	18	24	18	**60**
Summe	**29**	**42**	**29**	**100**

Die auf die Anforderungsbereiche verteilten Bewertungseinheiten innerhalb der Aufgaben sind als Richtwerte zu verstehen.